RANDOM NUMBER GENERATOR ARCHITECTURE AND VLSI IMPLEMETATION

N. Sklavos, P. Kitsos, K. Papadomanolakis and O. Koufopavlou.

VLSI Design Laboratory
Electrical and Computer Engineering Department
University of Patras, Patras, GREECE
email: nsklavos@ee.upatras.gr

ABSTRACT

Security protocols and encryption algorithms are basically based on random number generators. In this paper, a new random number generator architecture is introduced. The produced number word length is equal to 160 bits. The philosophy architecture relies on the usage of the SHA hash function. The offered security strength of this certain hash function ensures the unpredictability of the produced number. Additionally, an efficient VLSI implementation for FPGA device of the proposed system is described. The proposed architecture is a flexible solution in application cases where the original physical sources of random number generators, such as electrical noise, are not available or at least not convenient. This architecture can also be used in any cryptographic algorithm and encryption/decryption system with high-speed performance.

1. INTRODUCTION

The increasing demand of protecting wireless communications and transmitted information has forced researchers and system designers to develop more complicated and specialized cryptographic systems**.** In order these systems to withstand to the external attacks and to offer high level of security strength are basically based in random numbers. Cryptographic algorithms such as RSA, RC4 and protocols such as SET and SSL need random number generators in the initialization mode of their operation [1]. Random numbers are mainly used to generate public and private keys for the encryption/decryption algorithms. They are also applied in padding bytes, to salt passwords and random challenges in authentication protocols, such as Kerberos.

Random numbers are very difficult to be generated, especially by using deterministic systems. In many cases pseudorandom numbers are used instead of real random numbers. In such cases an observer can hardly distinguished between real random and pseudorandom numbers.

The most important issue in random number generators is that attackers, including those who know the random number generator architecture, must not be able to make useful predictions about the present and especially future produced random numbers. Existing designs of random number generators have flaw errors and in some cases system attackers have managed to analyze and recover the generation process of the produced random numbers [2].

In this work a new random number generator is presented. The produced random number has a word length equal to 160 bits. This length value is acceptable in all the random numbers applications.

The proposed architecture, which is based in the Secure Hash Algorithm (SHA), as would be analyzed in details in the next paragraphs can implemented and integrated in high speed systems. The security strength and the advantages of the SHA hash function that the proposed architecture is relied on, ensures the unpredictability in the produced numbers.

A VLSI implementation for FPGA device of the proposed architecture is described. The whole design was captured in VHDL language and synthesized for a specific XILINX FPGA device.

The proposed architecture is a flexible solution in application where the original physical sources of random number generators, such as electrical noise, are not available or have to be avoided.

This paper is organized as follows: in section two an introduction to entropy is given. In the next section the SHA hash function is presented, while in section 4 the proposed architecture is analyzed in details. The VLSI implementations results are described in the section 5 and finally some important conclusions are given in the last section.

2. ENTROPY

2.1 General Terms

In the mid 1950's, Andrei Kolmogorov imported Shannon's probabilistic notion of entropy into the theory of dynamical systems [3] and showed how entropy can (sometimes) be used to prove whether two dynamical systems are non-conjugate (i.e., non-isomorphic) [4].

There is a basic principle in dynamical systems. According to this principle there are a great variety of ways to define "entropy" in the context of dynamical systems. Nonetheless, these quantities are all closely related to one another (typically, they agree in an appropriate limit).

These relations show that entropy can be regarded quite generally as a measure of:

- unpredictability,
- incompressibility,
- asymmetry,
- delayed recurrence.

Entropy can also be regarded as the fractal dimension of an appropriate compact set.

The theory of dynamical systems is a major mathematical discipline closely intertwined with all main areas of mathematics. Its concepts, methods and paradigms greatly stimulate research in many sciences and gave rise to the vast new area variously called applied dynamics, nonlinear science, or chaos theory.

The entropy H in a message can be described from the following formula [5]:

$$H = - K \sum_{1}^{n} p_i * \log p_i$$

where p_i is the probability of the state i out of n possible states, while K is described as an optional constant to provide units (1/log(2) bits).

When we have to describe a random number generator with n bit, p_i is equal to the probability that an output will equal i, with range $0 \le i \le 2^k$. In the case of a perfect random generator, all possible outcomes in the produced number values have to be equally appeared. In other words this means that for a real random number generator $p_i = 2^{-k}$. The total entropy of the output is equal to k bits.

2.2 Entropy in the proposed architecture

Our intention, as demonstrated throughout this paper, is the creation of a random number generator, which should present increased metric entropy in the derivation of non-correlated and random-statistical numbers. This generator will be based in a dynamical system with a topology that is imposed by the analogue nature of the simple sinusoidal voltage format of a common power supply line. Because of the simplicity of this design and the widely use of the power supplies, this novel architectural scheme can serve as a random number generator, in the cryptographic genetic code.

3. THE SHA HASH FUNCTION

The SHA hash function [1], was designed by NIST [6] for the use of the digital signature standard. The SHA algorithm's main operation is the conversion of an input message of length less than 512 bits, to an output message of 160-bit (message digest).

The main difference of a hash function compared to the common block ciphers is that operates only in one direction. This means that hash functions produce the output block, called message digest, of any initial value but it is not possible to produce the pre-value for a given hash function output (message digest). In other words the security of such an algorithm is ensured due to the computationally infeasibility to create an initial message which would corresponds to a given message digest.

4. PROPOSED ARCHITECTURE

4.1 Proposed Architecture Diagram

The proposed architecture is illustrated in Fig. 1 .Two are the basic parts of the proposed system: the initial conditions generator (ICG) and the SHA hash function.

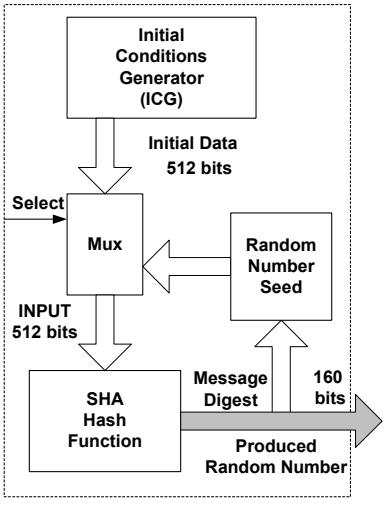

Figure 1. Proposed Architecture

The total system operation scenario is characterized of simplicity. In order to the SHA hash function to start operating for first time an initial data block with length range 40 to 512 is needed. This data block is produced by the initial conditions generator (ICG). After the first

operation of the SHA, the produced message digest is equal to the first produced random number. The message digest is kept in the seed.

In order to generate another random number, the SHA hash function operates again. The only difference in this time is that the appropriate initial data block is changed. In this time, a previous produced number from the random number seed is chosen. This different way in the selection of the initial data block is achieved with the use of a multiplexer (MUX) configured with the appropriate select signal (select). Finally, in this way, another random number is produced. The whole process goes on with the same way.

The selection of the SHA, as the operation algorithm core of the proposed architecture, has been done, considering security aspects.

The first major reason, is that the high security level of a hash function is ensured due to the computationally infeasibility to create an initial message which would corresponds to a given message digest. On the other hand hash functions could withstand to birthday attack attempts [1]. The selected hash function, SHA, is characterized of higher security strength compared with the other well know hash functions. This is another major advantage of the selected function.

4.2 Initial conditions generator (ICG)

The main purpose of the proposed ICG (Initial Conditions Generator) is to provide an initial data value that is derived from a non-deterministic procedure [7]. Actually, the mechanism realized is in fact a simple quite deterministic analogue mechanism. But, the source signal from which the initial conditions derive is based on a non-linear dynamic system with chaotic behavior.

The Initial Conditions Generator subsystem is based on a simple sinusoidal voltage source in conjunction with a DC voltage offset. It gives the initial signal for deriving the first obtained condition for the whole system. The simple architecture for this mechanism is shown in Fig. 2.

This mechanism can be implemented in almost every non-autonomous electric circuit with a common AC sinusoidal power supply voltage [8], and is inadequate for use in portable systems such as hand-held PCs and notebooks.

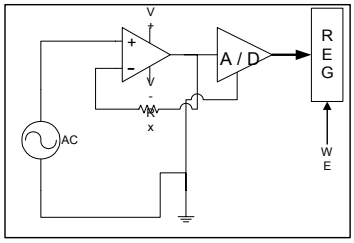

Figure 2. First Condition Implementation

While the behavior of the subsystem which id illustrated in Fig. 3 is quite deterministic, the initial time triggering for obtaining the first initial condition for the system is quite undetermined. This is due to is user dependant (based on the initiation timing of the whole cryptography scheme).

In order to further exploit the fortuity of the first condition, we use the derived value to define a time interval for triggering the data acquisition procedure used to form the total Initial Conditions value. This time interval is used to periodically obtain data values from the mechanism defined previously with a time period equal to the time interval derived from the first condition. If we consider the use of an 8-bit A/D converter and a 512 bit IC value, we need 64 time intervals in order to complete the data acquisition procedure. This forms the needed IC value by the random generator described later. An architectural scheme demonstrating this procedure is illustrated in Fig. 3.

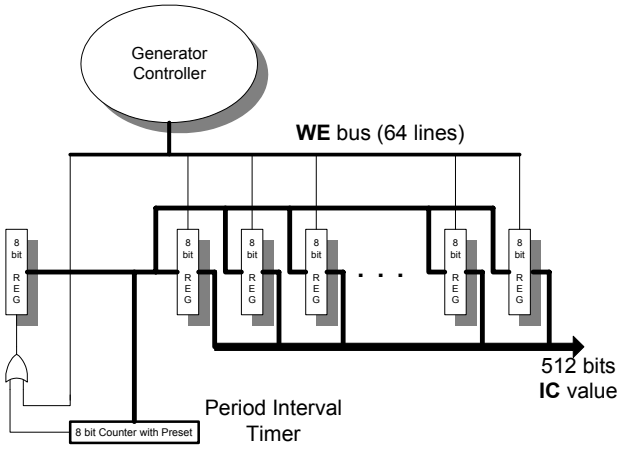

Figure 3. Initial Conditions Data Acquisition Mechanism

Although this procedure seems deterministic, and given the first condition value and an offset sinusoidal AC voltage supply could lead to determinable output signal, we haven't considered the fact that no perfect sinusoidal AC power voltage can be found. Because of the entropy that lies within the AC power supply signal, the 2nd data acquisition subsystem will lead to different IC output always even if we use the same first condition all the time.

The use of the two previously described subsystems can a priori give us an almost perfect random number generator circuit. Of course randomness is not the property required by this part of the whole architectural scheme. The needed property required is unpredictability, which also seems to be attained by the proposed implementation.

4.3 SHA Hash Function Architecture

The SHA hash function architecture is showed in Fig. 4. The algorithm basic round as the specifications of the standard defines have to be processed 80 times. In order to work properly a feedback loop in the basic round output has been added. An intermediate data register (named register) is used as temporary storage of the data after every transformation round. Finally, the data is transformed by the last transformation unit and in this way the message digest is produced. The two other components PADDING and WT_KT are fundamental parts of this certain hash function defined operation (for more details see [3]).

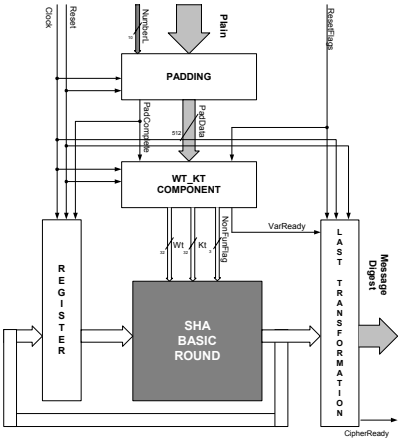

Figure 4. SHA Hash Function Architecture

4.3 Testing

The proposed architecture has been tested with the FIPS randomness tests [9]. No failure in these tests has been reported. The proposed architecture operation has been also tested and there was no error caused the output to "stuck" in one stage. Of course many different tests could be performed in such a design but according to our knowledge the executed test set detects the most important and commonly appeared failures.

5. VLSI IMPLEMENTATION RESULTS

The proposed architecture has been captured by using VHDL. All the internal components of the design were synthesized placed and routed using FPGA device of XILINX. The system was simulated for verification of the correct functionality in real time operating conditions. The test vectors provided by NIST [6] for the SHA algorithm were applied to assure that the design was working properly. The measurements of the synthesis results for this implementation are presented in table I.

FPGA DEVICE	XILINX v300pq240	
AREA ALLOCATION	**Used / Available**	**Utilization**
IOs	150 / 166	90 %
Fun. Generators	5212 / 6144	85 %
CLB Slices	2606 / 3072	85 %
Dffs or Latches	1560 / 6144	25 %
f (MHz)	37	

Table I. FPGA Implementation Characteristics

6. CONCLUSIONS

In this paper a new random number generator architecture and its VLSI implementation is presented. Cryptographic algorithms and communications protocols are based on random numbers generators. The security strength and the advantages of the selected SHA hash function ensure the unpredictability in the produced random numbers. The proposed architecture is a flexible solution in application cases where the original physical sources of random number generators are not available or have to be avoided.

7. REFERENCES

[1] Bruce Schneier, *Applied Cryptography – Protocols, Algorithms and Source Code in C*, Second Edition, John Wiley and Sons, New York, 1996.

[2] J. Kelsey, B. Schneier, D. Wagner and C. Hall, "Side Channel Cryptanalysis of Product Ciphers", ESORICS '98 Proceedings, Springer-Verlag, 1998.

[3] Anatole Katok and Boris Hasselblatt, "Introduction to the Modern Theory of Dynamical Systems", Cambridge University Press, 1995.

[4] Edward Ott, Tim Sauer, and James A. Yorke, "Coping with Chaos", Wiley-Interscience, 1994.

[5] C.E. Shannon, "A mathematical theory of communication", The Bell System Technical Journal, vol. 27, July 1948.

[6] National Institute of Standards and Technology, Secure Hash Standard, www.itl.nist.gov/fipspubs/fip180-1.htm.

[7] R. Caponetto, G. Di Bernardo, E. Di Cola, L Occhipinti, "A New Chaotic System for the Authentication and Electronic Certification Procedures", ICECS 1999, pp. 1235-1238.

[8] I. M. Kyprianidis, P. Haralabidis, I. N. Stouboulos, "Controlling and Synchronization of a Second-Order Non-Autonomous Nonlinear Electric Circuit", ICECS 1999, pp. 1247-1251, Sep. 1999.

[9] Federal Information Processing Standards Publication 140-1, "Security Requirements for Cryptographic Modules", U.S. Department of Commerce/NIST, Springfield, VA : NTIS, 1994.